WHAT IS CLAIMED IS:

1. An information processing method for calculating x*(2^n) mod P for an input value x larger than a prime number P, the operator ^ denoting power, wherein:

the value x*(2^n) mod P is calculated without explicitly obtaining x mod P, by:

calculating or previously preparing 2^(2m+n) mod P when the input value x has to be transformed into x*(2^n) mod P, the number n denoting the number of bits necessary and sufficient for storing the modulus P and the number m denoting the number of bits necessary for storing the input value x;

calculating x1 = x*2^(2m+n)*(2^(-m)) mod P = x*2^(m+n) mod P by Montgomery modular multiplication; and

calculating x2 := x1*(2^(-m)) mod P = x*(2^n) mod P.

2. An information processing method for calculating x*(2^n) mod P for an input value x larger than a prime number P, the operator ^ denoting power, wherein:

the value x*(2^n) mod P is calculated without explicitly obtaining x mod P, by:

calculating or previously preparing 2^(m+2n) mod P when the input value x has to be transformed into x*(2^n) mod P, the number n denoting the number of bits necessary and sufficient for storing the modulus P and

the number m denoting the number of bits necessary for storing the input value x;

calculating x1 = x*2^(m+2n)*(2^(-m)) mod P = x*2^(2n) mod P by Montgomery modular multiplication; and

calculating x2 := x1*(2^(-n)) mod P = x*(2^n) mod P.

3. An information processing method for conducting a modular exponentiation operation x^d mod P for an input value x and an exponent d, by combining results of exponentiation operations each of which is carried out for each s-bit segment successively extracted from the exponent d, wherein:

the value x^d mod P is calculated not by calculating x^d[i] mod P, the exponent d[i] denoting i-th segment of the extracted s-bit segment of the exponent d, but by:

calculating (2^n)^(2^n-1)*x^d mod P by use of (2^n)^(2^s-1)*x^d[i] mod P, the number n denoting the number of bits necessary and sufficient for storing the modulus P and the number m denoting the number of bits necessary for storing the input value x; and

calculating the value x^d mod P by multiplying the above result (2^n)^(2^n-1)*x^d mod P by 2^(-n)^(2^n-1) mod P.